Faculdade de Engenharia da Universidade do Porto

MINIX CHESS

Projeto final de Laboratório de Computadores

Turma 7 - Grupo 5
Alexandre Abreu Filho
up201800168
João Castro Pinto
up201806667

Porto, Janeiro de 2020

Índice

1. Instruções de utilização do programa
    1.1 Menu inicial
    1.2 Jogo
        1.2.1 Objetivos
        1.2.2 Peças
            1.2.2.1 Rei
            1.2.2.2 Rainha
            1.2.2.3 Torre
            1.2.2.4 Bispo
            1.2.2.5 Cavalo
            1.2.2.6 Peão
        1.2.3 Roque
        1.2.4 Modos
            1.2.4.1 Modo normal
            1.2.4.2 Modo “blindfold”
            1.2.4.3 Modo “hot seat”
    1.3 Ecrã final
2. Estado do projeto
    2.1 Dispositivos utilizados
        2.1.1 Temporizador
        2.1.2 Teclado
        2.1.3 Rato
        2.1.4 Placa gráfica
        2.1.5 RTC
        2.1.6 Portas de série

3. Estrutura do código
- 3.1 Módulos
  - 3.1.1 Timer
  - 3.1.2 KBC
  - 3.1.3 Keyboard
  - 3.1.4 Mouse
  - 3.1.5 Graphics
  - 3.1.6 Draw
  - 3.1.7 Chess
  - 3.1.8 Game
  - 3.1.9 Arrow
  - 3.1.10 Interrupts
  - 3.1.11 Menu
  - 3.1.12 RTC
  - 3.1.13 Macros
  - 3.1.14 Art
  - 3.1.15 Serial
  - 3.1.16 Main
- 3.2 Gráfico de chamada de funções

4 Detalhes de implementação

5 Conclusões

1. Instruções de utilização do programa

1.1 Menu inicial

No menu inicial são apresentadas quatro opções ao utilizador, nomeadamente, carregar a tecla “n” para iniciar um jogo no modo normal, a tecla “b” para iniciar um jogo no modo “*blindfold*”, a tecla “s” para iniciar um jogo no modo “*hot seat*”. Por último, a tecla “esc” é utilizada para sair do jogo, em qualquer momento do programa.

Figura 1 - Menu inicial.
Fonte: elaborada pelo autor.

1.2 Jogo

1.2.1 Objetivos

Cada jogador controla as peças de uma cor e tem como objetivo capturar o rei inimigo, o jogo é baseado em turnos, e as brancas começam sempre. Jogadas que deixem o próprio rei em risco não são permitidas.

O jogo acaba quando um dos jogadores desiste, sai do jogo ou o jogador do turno fica sem movimentos possíveis, sendo que o último termina em empate (“*stalemate*”) se o rei não estiver ameaçado e em vitória (“*checkmate*”) se o rei estiver ameaçado.

1.2.2 Peças

Cada peça tem um design e movimentos diferente. Um movimento só é possível se não existirem peças nas casas que ficam entre as casas inicial e final do mesmo, exclusive, à exceção do cavalo, que “salta” por cima de outras peças.
Todas as peças capturam uma peça que esteja no final de seu movimento, com exceção do peão que tem movimentos diferentes se houver capturas ou não.
Quando um peão chega à última (peão branco) ou à primeira (peão preto) linha do tabuleiro ele é promovido e se transforma em uma rainha.

1.2.2.1 Rei

O rei é a peça mais valiosa do jogo e pode mover-se em qualquer direção mas apenas uma casa por jogada, como mostra a figura 2.

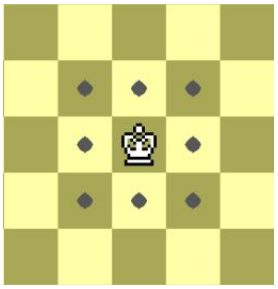


Figura 2 - O rei e seus movimentos.
Fonte: Elaborada pelo autor.

1.2.2.2 Rainha

A rainha é a peça com mais movimentos do jogo e pode mover-se em qualquer direção como o rei. Tem a vantagem de se poder mover quantas casas forem necessárias, como mostra a figura 3.

Figura 3 -
A rainha e seus movimentos.
Fonte: Elaborada pelo autor.

1.2.2.3 Torre

A torre move-se verticalmente ou horizontalmente, quantas casas quanto forem necessárias, como mostra a figura 4.

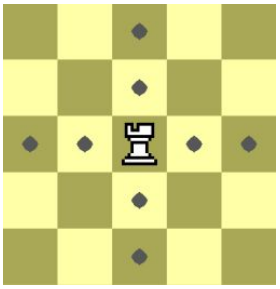


Figura 4 - A torre e seus movimentos.
Fonte: Elaborada pelo autor.

1.2.2.4 Bispo

O bispo move-se diagonalmente, quantas casas forem necessárias, como mostra a figura 5.

Figura 5 -
O bispo e seus movimentos.
Fonte: Elaborada pelo autor.

1.2.2.5 Cavalo

O cavalo move-se “como um L”, duas casa numa direção das linhas e colunas do tabuleiro e uma casa na direção perpendicular à primeira, como mostra a figura 6. Além disso, o cavalo é a única peça que pode “saltar” outras peças, ou seja, seus movimentos são válidos mesmo que haja peças no caminho.

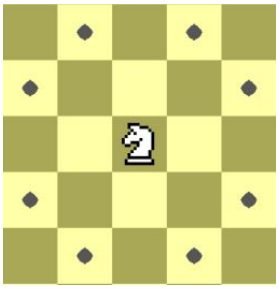


Figura 6 -
O cavalo e seus movimentos.
Fonte: Elaborada pelo autor.

1.2.2.6 Peão

O peão tem movimentos diferentes em situações diferentes. Independentemente de qual seja o movimento, a peça anda sempre um casa para a frente (aumenta o número da linha para os brancos e diminui para os pretos).
O movimento padrão do peão é avançar uma casa em frente como mostra a figura 7 (se for o primeiro movimento da peça pode-se mover duas casas também, como mostra a figura 8).
Se houver captura, o peão avança uma casa a frente e uma à direita ou esquerda, como mostra a figura 9.

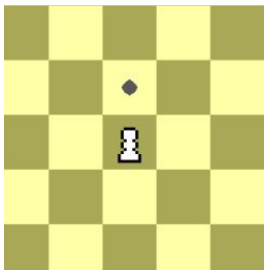


Figura 7 -
O movimento padrão do peão.
Fonte: Elaborada pelo autor.

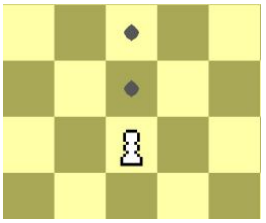


Figura 8 -
O movimento do peão que ainda não moveu-se.
Fonte: Elaborada pelo autor.

Figura 9 -
O movimento de captura do peão.
Fonte: Elaborada pelo autor.

1.2.3 Roque

Além dos movimentos acima explicados existe um movimento especial chamado roque. Ele consiste em mover o rei duas casas a direita ou à esquerda e mover a torre para uma casa ao lado do rei na direção contrária ao movimento do último.

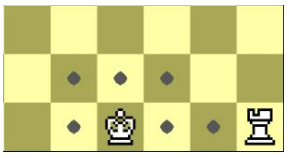


Figura 10 - Antes do roque do lado do rei
Fonte: Elaborada pelo autor.

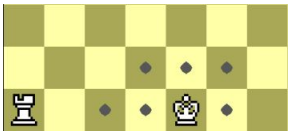


Figura 11 - Antes do roque do lado da rainha
Fonte: Elaborada pelo autor.

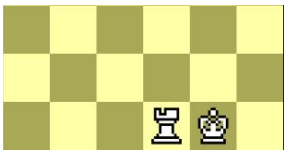


Figura 12 - Depois do roque do lado do rei
Fonte: Elaborada pelo autor.

Figura 13 - Depois do roque do lado da rainha
Fonte: Elaborada pelo autor.

1.2.4 Modos

1.2.4.1 Modo normal

Este modo apenas funciona com porta de série.

Quando se clica com o botão esquerdo do rato numa peça do tabuleiro, as casas que correspondem a movimentos possíveis serão marcadas com um círculo cinzento, e ao clicar numa dessas casas marcadas, o movimento é realizado.

Assim que o jogo é terminado, aparece o ecrã final, isto é, explica a razão de o jogo ter acabado e ao carregar na tecla "*esc*", o programa é terminado.

Além disso, é apresentado um temporizador de cada lado do tabuleiro (esquerdo do jogador de peças brancas e direito do jogador de peças pretas) para indicar quanto tempo cada jogador ainda tem para jogar. Se um jogador esgotar o tempo, este perde o jogo.

A qualquer momento podem ser carregadas as teclas "r" para desistir do jogo e "esc" para terminar o jogo. Como o modo *"blindfold" também utiliza porta de série,* o modo normal pode ser emparelhado com o modo *"blindfold"*. Embora não apareça o tempo restante de cada jogador no ecrã de *"blindfold"* quando se joga com os modos emparelhados, o tempo é apresentado no ecrã do modo normal.

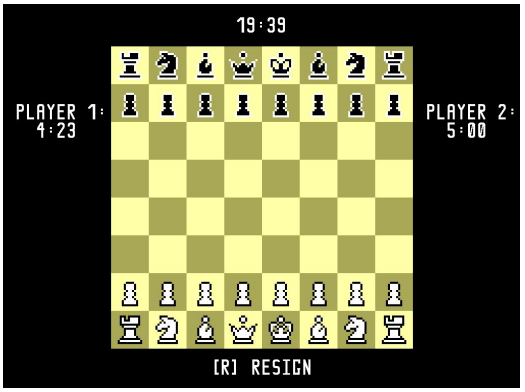


Figura 14 - Tabuleiro de jogo
Fonte: Elaborada pelo autor

1.2.4.2 Modo "*blindfold*"

As regras do modo "blindfold" são as mesmas que do modo normal, porém não há interface gráfica mostrando o tabuleiro com as peças, os jogadores tem que ter o tabuleiro na memória. Neste modo cada jogador digita o seu movimento na forma <coluna inicial><linha inicial><coluna final><linha final>, sendo as colunas representadas por letras de "a" a "h" e as linhas por números de "1" a "8". O outro jogador terá essa jogada apresentada em seu ecrã e terá de digitar sua jogada.
Este modo apenas funciona com porta de série.
A qualquer momento podem ser carregadas as teclas "r" para desistir do jogo e "esc" para fechar o jogo.

Como o modo normal *também utiliza porta de série,* o modo *"blindfold"* pode ser emparelhado com o modo normal. Embora não apareça o tempo restante de cada jogador no ecrã de *"blindfold"* quando se joga com os modos emparelhados, o tempo é apresentado no ecrã do modo normal.

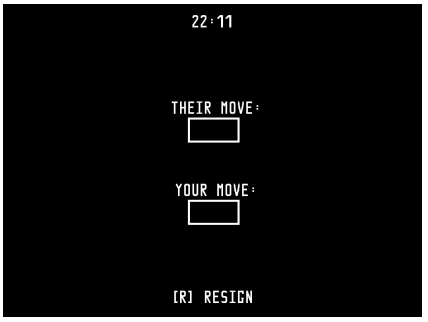


Figura 15 - Ecrã modo "blindfolded"
Fonte: Elaborada pelo autor

1.2.4.3 Modo "*hot seat"*

Este modo funciona da mesma forma que o modo normal, mas com a particularidade de não utilizar a porta de série. O jogador controla as duas cores na mesma máquina.

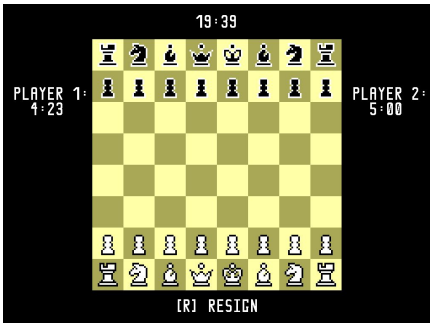


Figura 16 - Tabuleiro "hot seat"
Fonte : Elaborada pelo autor

1.3 Ecrã final

Quando o jogo acaba, os jogadores são enviados a um último ecrã que indica como o jogo acabou, e pode-se terminar o programa nesse momento com a tecla "esc".

2. Estado do projeto

2.1 Dispositivos utilizados

Dispositivo	Utilização	Interrupções
Temporizador	Contar os tempos de cada jogador	Sim
Teclado	Teclas de ação e movimentos no modo *“blindfold”*	Sim
Rato	Movimento das peças	Sim
Placa gráfica	Interface gráfica do jogo	Não
RTC	Apresentação da hora atual	Não
Portas de série	Sincronização de jogo entre dois jogadores	Sim

2.1.1 Temporizador

No jogo de xadrez cada jogador tem um tempo limite para pensar em suas jogadas, o principal objetivo do temporizador é contar quanto tempo passou-se em cada jogada para que o jogador que tenha seu tempo esgotado perca o jogo.
Além disso, as funções do RTC são chamadas de acordo com as interrupções do temporizador. Este protocolo é descrito na secção do RTC.

2.1.2 Teclado

O teclado é utilizado no menu para selecionar o modo (“n” para o modo normal e “b” para o modo *“blindfold”*) ou sair do jogo com a tecla “ESC” em qualquer momento, inclusive no menu.
Também é utilizado para atalhos durante o jogo (“r” para desistir e “d” para pedir um empate) e para receber o input dos movimentos no modo *“blindfold”* em formato de texto.

2.1.3 Rato

O rato é o principal dispositivo para o jogador no modo normal e *“hot seat”*, a posição do rato é utilizada pois o programa desenha sempre um sprite da seta do rato sempre a cada três interrupções do mesmo (uma vez que cada pacote do rato tem três bytes, por cada byte que e o rato envia, é gerada uma interrupção) e os movimentos são feitos com um clique no botão esquerdo na posição inicial da peça

a mover, e outro na posição final. Repare-se que foi utilizada uma máquina de estados muito simples, com apenas dois estados.

2.1.4 Placa gráfica

Utiliza-se a placa gráfica no modo 105h, com uma resolução de 1024x768, com cores indexadas e configuração de 64 cores.
Além disso, a placa gráfica é essencial no modo normal de jogo, onde são utilizados três buffers, o primeiro tem apenas o tabuleiro com as peças, o segundo buffer tem, além do que estiver no primeiro, todas as outras coisas com exceção do rato (círculos que indicam os movimentos possíveis, tempos de cada jogador, horário atual, etc), e o terceiro, que é a própria memória de vídeo, tem tudo o que estiver no segundo mais o rato. Portanto, cada buffer atua como um layer que copia pixels do layer abaixo quando remove um elemento.
Não há colisões no jogo mas há sprites animados, o rato. A fonte utilizada no projeto foi criada por nós especialmente para o projeto.

2.1.5 RTC

O RTC é utilizado para ler a hora atual para poder ser escrita no ecrã ao longo de todo o programa. Embora o código para levantamento de interrupções por alarme, tenha sido implementado, não conseguimos que as interrupções fossem geradas para atualizar a hora. Portanto, optou-se por recorrer ao timer. Até ocorrer ao minuto mudar desde o início do programa, sempre que há uma interrupção do timer lê-se a hora do rtc. Quando ocorrer uma mudança do minuto, deixa-se de ler a hora do rtc a todos os segundos e passa-se a ler a hora a cada minuto. Só se atualiza a hora no ecrã quando há uma mudança da hora. A função que é utilizada para ler a hora é “rtc_read_time” e para escrever no ecrã é “write_string”.

2.1.6 Portas de série

A porta de série é utilizada no programa para o modo normal e *“blindfold”*. Aqui foi implementada uma máquina de estados para descobrir quando os dois jogadores estão prontos a começar o jogo. Quando um jogador seleciona um dos modos que utiliza porta de série, envia ao computador na outra linha da porta de série um caráter que indica que está à espera do segundo jogador. O segundo jogador utiliza este sinal para saber se pode passar ao jogo, e responde ao primeiro jogador. Quando este responde, ambos os computadores entram no jogo. No jogo, quando se tem um movimento completo, seja no modo normal ou no modo *“blindfold”*, este é enviado para o outro jogador. Cada movimento é um string de 4 caracteres no formato <coluna inicial><linha inicial><coluna final><linha final>, em que os primeiros dois especificam a posição inicial da peça a mover e os últimos dois . Antes de enviar o movimento, este é processado pelo jogo no computador que envia, e o computador que recebe o movimento processa-o logo que o recebe.

3. Estrutura do código

3.1 Módulos

3.1.1 Timer

Este módulo lida com interrupções, estado e valor do temporizador. Foram refeitas as funções do lab 2.
Desenvolvido por Alexandre Abreu (50%) e João Pinto (50%) e vale 6% do projeto.

3.1.2 KBC

Este módulo lida com os acessos aos registos do kbc. Foram refeitas as funções dos labs 3 e 4.
Desenvolvido por Alexandre Abreu (50%) e João Pinto (50%) e vale 3% do projeto.

3.1.3 Keyboard

Este módulo lida com interrupções do teclado e armazenamento do último scancode. Foram refeitas as funções do lab 3.
Desenvolvido por Alexandre Abreu (50%) e João Pinto (50%) e vale 5% do projeto.

3.1.4 Mouse

Este módulo lida com interrupções e armazenamento dos últimos pacotes recebidos. Foram refeitas as funções do lab 4.
Desenvolvido por Alexandre Abreu (50%) e João Pinto (50%) e vale 6% do projeto.

3.1.5 Graphics

Este módulo lida com as configuração da placa gráfica, da memória de vídeo e da implementação de triple buffering. Foi inicialmente feito no lab 5.
Desenvolvido por Alexandre Abreu (70%) e João Pinto (30%) e vale 4% do projeto.

3.1.6 Draw

Este módulo foi inicialmente desenvolvido para o lab 5. Contém as funções principais para desenhar no ecrã, desde a função draw_pixel que desenha um pixel no ecrã, até à função write_string que desenha strings no ecrã.
Desenvolvido por Alexandre Abreu (50%) e João Pinto (50%) e vale 12% do projeto.

3.1.7 Chess

Este módulo lida com o xadrez, cria e administra o estado do tabuleiro, verifica e realiza as jogadas. O tabuleiro é uma matrix 8x8 de apontadores para peças que são structs com dois atributos (tipo e cor) alocadas dinamicamente.
Desenvolvido por Alexandre Abreu (100%) e vale 17% do projeto.

3.1.8 Game

Este módulo faz a interface do jogo com periféricos utilizados e entre os periféricos, por exemplo escrever o tempo restante de cada jogador no ecrã. Neste módulo está implementado também o desenho do que é estático do modo blindfold.
Desenvolvido por Alexandre Abreu (50%) e João Pinto (50%) e vale 7% do projeto.

3.1.9 Arrow

Este módulo é utilizado para imprimir o cursor do rato no modo normal e hot seat no ecrã. Repare-se que a escrita do rato no ecrã, está otimizada para não escrever mais do que deve.
Desenvolvido por Alexandre Abreu (50%) e João Pinto (50%) e vale 6% do projeto.

3.1.10 Interrupts

Este módulo lida com subscrições de dispositivos e chamadas às funções do ciclo de interrupções para deixar o ficheiro main.c com uma formatação mais simples.
Desenvolvido por Alexandre Abreu (100%) e vale 4% do projeto.

3.1.11 Menu

Neste módulo encontram-se as implementações do menu, do ecrã de espera que outro jogador se conecte e do ecrã final.
Desenvolvido por Alexandre Abreu (20%) e João Pinto (80%) e vale 8% do projeto.

3.1.12 RTC

Neste módulo foram desenvolvidas as funções para lidar com o RTC. Repare-se que o código para geração de interrupções por alarme foi implementado, embora não tenha sido implementado. O último tempo lido, é guardado numa string no formato “hh:mm”.
Desenvolvido por João Pinto (100%) e vale 2% do projeto.

3.1.13 Macros

Este módulo contém todas as macros necessárias em outros ficheiros e inclui as macros utilizadas nos labs.
Desenvolvido por Alexandre Abreu (50%) e João Pinto (50%) e vale 1% do projeto.

3.1.14 Art

Este módulo contém todas as imagens utilizadas no programa. As peças foram inspiradas pelo design de peças pixelizadas do site lichess.com.
Desenvolvido por Alexandre Abreu (35%) e João Pinto (65%) e vale 1% do projeto.

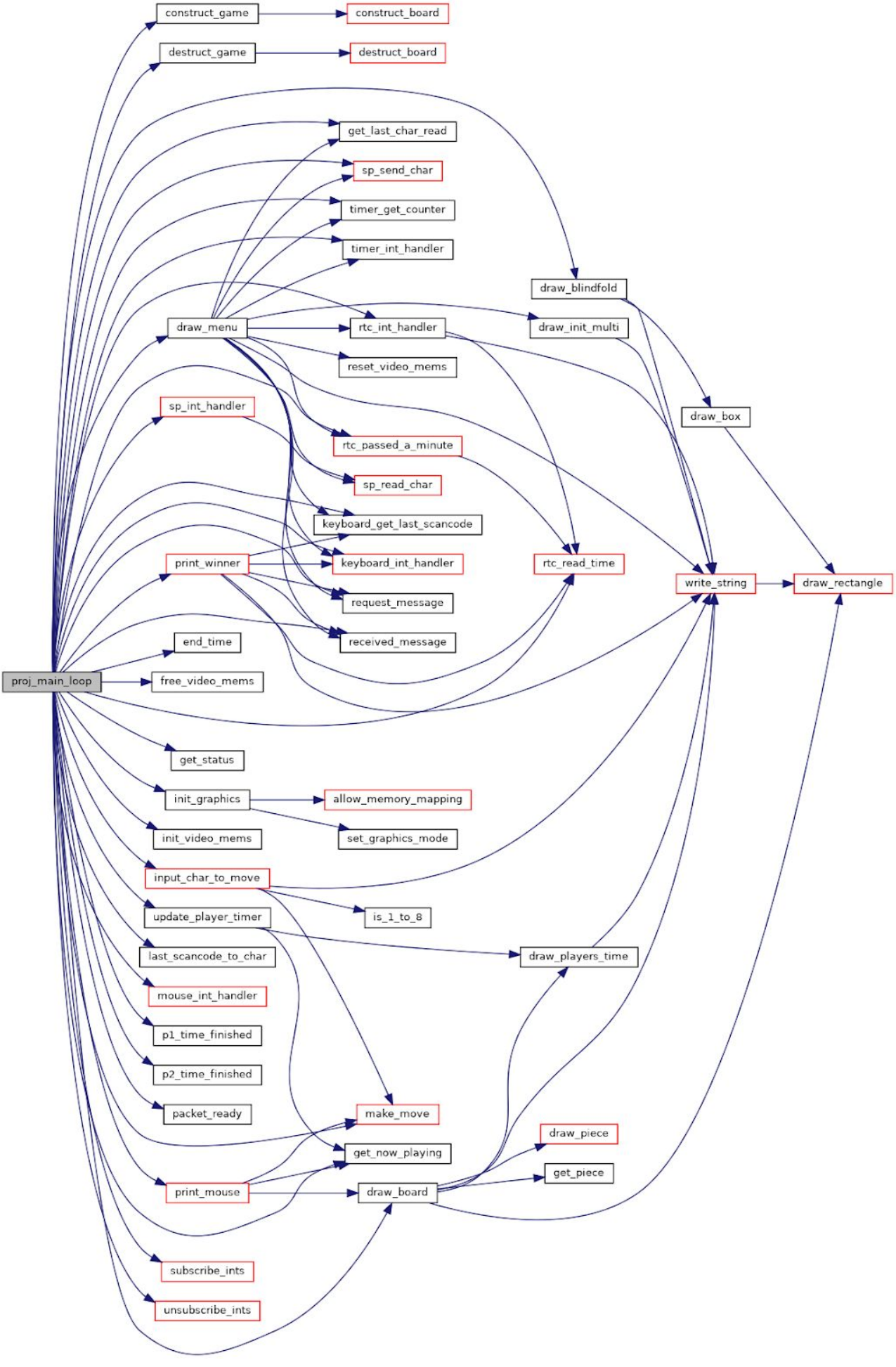
3.1.15 Serial

Este módulo lida com interrupções e envio e recepção de dados. Este permite que sejam armazenados os quatro últimos caráteres com a função sp_store_last_char.
Desenvolvido por Alexandre Abreu (30%) e João Pinto (70%) e vale 8% do projeto.

3.1.16 Main

No módulo main é onde fica o ciclo de interrupções, são chamadas os módulos principais do projeto e contém os diferentes modos de jogo.
Desenvolvido por Alexandre Abreu (50%) e João Pinto (50%) e vale 10% do projeto.

3.2 Gráfico de chamada de funções



4 Detalhes de implementação

Foi decidida a utilização de três buffers, e do desenho do rato diretamente na memória de vídeo, pois isso foi otimizado a ponto de acharmos não valer a pena criar mais um buffer para o movimento do rato.
Foi criada uma função que escreve strings no modo gráfico, o que ajudou a fazer todos os menus e a interface do modo blindfolded.
Na maioria dos módulos tentou-se utilizar o conceito de orientação a objetos em C, principalmente nos módulos Chess, Mouse, Timer e Keyboard, que foi muito interessante pois permitiu que aprendêssemos algo que tínhamos idéia de ser muito difícil.
O xadrez é um jogo com poucas cores, por isso foi escolhido o modo 105h, pois economiza processamento e memória.

5 Conclusões

Consideramos que esta unidade curricular foi bastante boa, no sentido em que promove o ensino autosuficiente. Embora as aulas teórico-práticas tenham ajudado na realização do projeto e dos labs, a maior do parte do trabalho foi feito fora de tempo de aulas.
A matéria tem um nível de dificuldade não condizente com a carga horária, pois outras matérias com carga igual ou até superior necessitaram de menos tempo.
Pode-se verificar que o sistema operacional utilizado em aulas, Minix, não tem muito valor prático, já que é muito pouco usado.
Os professores das aulas práticas puderam nos ajudar com a maioria das dúvidas que tiramos em seus gabinetes.